Remarks

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 1-10 and 12-25 are pending in the application. Claims 1-10 and 12-25 are rejected. No claims have been allowed. Claims 1, 3, 6, 7, 14, 18, and 23 are independent. Claims 1, 6, 7, 14, 18, and 23 have been amended.

Cited Art

The Action cites Youfeng Wu, US 7,032,217 (hereinafter "Wu"), Alexander et al., US 6,658,652 (hereinafter "Alexander"), and Zorn et al., "A Memory Allocation Profiler for C and Lisp Programs" (hereinafter "Zorn").

Claim Rejections - 35 U.S.C. § 112

The Action rejects Claims 6, 7-13, and 18-25 under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

With respect to claim 6, the Action rejects the claim for reciting the language "the frequency" with insufficient antecedent basis. Applicants have corrected the claim to recite "a frequency." Additionally, the Action rejects claim 6 stating "[i]t is also not clear what/how to cause the frequency decreases." Applicants respectfully disagree that the previous claim language was unclear, but to expedite prosecution Applicants have amended claim 6 to recite "said frequency at which the bursts are performed comprising the number of executions of the copy of the procedure divided by the total number of executions of both the original procedure and the copy of the procedure." Applicants believe that, with these amendments, claim 6 is now in condition for allowance.

With respect to claim 7, the Action rejects the claim, stating that "[i]t is not clear how to cause the instrumented versions to be sampled at a higher rate," as well as "[i]t is also not clear about the definition of 'rate' as amended. While Applicants respectfully disagree that the previous claim language was unclear, to expedite prosecution, Applicants have amended claim 7 as follows:

> ...executing the instrumented version of the software, wherein the instrumented version of the procedures are *sampled at higher rates for procedures whose total number of executions of both the original versions and the copy versions are executed less frequently and sampled at lower rates for procedures whose total number of executions of both the original versions and the copy versions are executed more frequently*, wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as *a percentage of total number of executions of both versions of the procedure*;

[Emphasis added.] Applicants believe that, with these amendments, claim 7 is now in condition for allowance.

The Action also rejects independent claims 18 and 23 for similar reasons. Applicants respectfully note that claim 18, as amended, now recites, "sampling a copy of a procedure at a rate inversely proportional to how frequently both the original and the copy of the procedure are executed." Claim 23 now recites, "sampling a copy of a procedure at higher rates for procedures whose original versions and copies are executed less frequently and sampling a copy of a procedure at lower rates for procedures whose original versions and copies are executed more frequently." Applicants believe that, with these amendments, claims 18 and 23 are now in condition for allowance.

Claims 8-13, 19-22, 24, and 25, each depend from one of independent claims 7, 18, and 23. As such, Applicants believe that, with these amendments, the dependent claims are in condition for allowance. Applicants respectfully request that the rejection of claims 6-10, 12, 13 and 18-25 be withdrawn and that the claims be allowed.

Double Patenting Rejections

The Action rejects claims 1, 6, 18, and 23 under nonstatutory obviousness-type double patenting as being unpatentable over claim 1 of Chilimbi. Applicants respectfully submit the claims, as amended, are patently distinct over the cited art.

In its rejections, the Action appears to allege that the claims are each unpatentable over claim 1 alone and with no modification. [*See*, Action, at § 8, pages 6-9, where only the language of claim 1 of Chilimbi is recited against the instant claims]. For this reason, Applicants assume the Action intends to allege the claims are anticipated by Chilimbi.

Applicants respectfully submit the claims in their present form are allowable over the cited art. For an anticipation rejection to be proper, the cited art must show each and every element as set forth in a claim. However, Chilimbi does not describe each and every element.

Claims 1, 18, and 23

As Applicants noted previously, instant claims 1, 18, and 23 each recite some form of adjusting or changing sampling rates. For example, claim 1 recites:

> adapting the sampling rate for the code paths according to the frequency of execution of the code paths, such that, after adapting, a ratio of a number of executions of the instrumented version code path to a total number of executions of the code path is equivalent to the adjusted sampling rate.

[Emphasis added.] Claim 18 recites:

> sampling a copy of a procedure at a rate inversely proportional to how frequently both the original and the copy of the procedure are executed;

[Emphasis added.] And claim 23 recites:

> sampling a copy of a procedure at higher rates for procedures whose original versions and copies are executed less frequently and sampling a copy of a procedure at lower rates for procedures whose original versions and copies are executed more frequently

Examples of sampling rates, and how they are adjusted, are found in the Application, for example starting at page 6:

> In bursty tracing with adaptive instrumentation, this sampling rate is adapted to the frequency of execution of the code path through the adaptive dispatch check. The more often the code path (i.e., the adaptive dispatch check) is executed, the more the sampling rate is decreased. In one implementation, all adaptive dispatch checks initially produce bursty trace samples at a rate at or near 100% (full tracing). . . .
>
> . . .
>
> []For example, in one implementation, the sampling rate is decremented by a factor of 10 each time the sampling rate is decreased, e.g., from 100%, 10%, 1%, 0.1%, etc. The interval determines how often to decrement the sampling rate. In one implementation, the sampling rate is decremented progressively less often. For example, the interval between decrements can be increased by a factor of 10

> each time time the sampling rate is decremented, e.g., from an interval of 10 nCheck counter resets, to 100, 1000, 10,000, etc. The bound counter determines the lower bound of the sampling rate for the adaptive dispatch check.
>
> In each case, the Action cites either "tracking a number of iterations" or "the tracked number of iterations."

Hence, the claims recite *changing* rates of sampling.

In the Action's rejection of this language of claims 1, 18, and 23, it cites to the following language from claim 1:

> *alternately tracking a number of iterations* of the check code executed in a checking phase and a profiling phase *up to respective checking and profiling count parameters, . . . and switching between checking and profiling phases upon the tracked number of iterations of the check code reaching the respective count parameter* of the respective phase.

[Action, at pages 7 and 9.]

Applicants first note that, in the response to the Arguments section of the Action, the Action that "[t]he 'sampling rate' as in a form of ratio or percentage is just a simple calculation of Chilimbi's profiling count parameter divided by a number of iterations." [Action, at page 3.] However, since claim 1 of Chilimbi recites switching between checking and profiling based entirely on the count parameters, it is these count parameters in Chilimbi that define this ratio. For example, for a checking count parameter of 5 and a profiling count parameter of 3, the method of claim 1 of Chilimbi would necessarily switch to profiling after 5 iterations and back to checking after 3 more. This would define a profiling ratio of 5/8. Thus, any analog in Chilimbi to the "sampling rate" would be defined by the values of these count parameters.

Applicants respectfully note, however, that because of this, *the rate at which sampling is done in claim 1 of Chilimbi does not change.* Chilimbi makes no mention of changing either count parameter in claim 1. As such, the count parameters are not described as changing, and therefore the rate defined by them is not described as changing either. Applicants also argue that because the ratio is unchanging, claim 1 of Chilimbi teaches away from adjusting it as well. Therefore, claim 1 of Chilimbi does not read upon, teach, or suggest, the above-quoted "adjusting" language of the claims.

Claim 6

Claim 6, which is also rejected in the Action, recites:

> executing the executable version of the program, wherein the copies of the procedures are executed in bursts, *and the frequency at which the bursts are performed decreases as the total number of executions of either the original procedure or copy of the procedure is executed*, said frequency comprising the number of executions of the copy of the procedure divided by the total number of executions.

[Emphasis added.] Again, similarly to the "adjusting" argument above, Applicants note that claim 1 of Chilimbi does not recite any change in frequency of performance of bursts. The Action appears to find this language in the "switching" language of claim 1 of Chilimbi discussed above, as well as in the "upon executing the check code" language, which simply describes the switching between non-instrumented and instrumented versions of code as described above. Thus, for similar reasons to those above, claim 1 of Chilimbi does not describe, nor does it teach or suggest, the above-quoted language of instant claim 6.

For at least these reason, the Action fails to make a establish proper case of nonstatutory double patenting over claim 1 of Chilimbi. Accordingly, applicants request that the double patenting rejection be withdrawn.

Claim Rejections - 35 U.S.C. § 103(a)

The Action rejects claims 1-6, 14, 18, 19, and 23 under 35 U.S.C. § 103(a) as unpatentable over Wu. The Action also rejects claims 7-13 and 15-17 under 35 U.S.C. § 103(a) as unpatentable over Wu in view of Alexander. The Action also rejects claims 20-22 and 24-25 under 35 U.S.C § 103(a) as unpatentable over Wu in view of Zorn.

Applicants respectfully submit that a prima facie case has not been made that the claims are unpatentable. In particular, Applicants note that the cited art, even as modified as suggested in the Action, fails to teach or suggest every aspect of the claims. Accordingly, applicants request that all rejections be withdrawn. Claims 1, 3, 6, 14, 18, and 23 are independent.

Claim 1

Claim 1, as amended, recites, in part:

> *creating an instrumented version of the program comprising duplicate versions of at least some code paths in the programs, such that a duplicate code path has an original version code path and an instrumented version code path* with instrumentation code for capturing instrumentation data;
>
> . . .
>
> *adapting the sampling rate for the code paths* according to the frequency of execution of the code paths,

[Emphasis added.] The Application, early in its Detailed Description, describes the use of duplicate code paths, as well as adaptive switching between the two paths:

> A bursty tracing framework is described briefly in reference to FIG. 1. FIG. 1 is a block diagram of a program modified according to a bursty tracing framework 100 . *Original procedures 110 in the target software are duplicated such that a checking code (original code) 120 is produced along with an instrumented code 130.* The instrumented code 130 can contain any number of instrumentation points. The framework 100 can comprise dispatch checks 140-141, which can be placed, for example, at procedure entry and loop back-edges. *The dispatch checks are responsible for diverting control between the checking and instrumentation copies of the code based on counters relating to sample size (nInstr) and sampling period or rate (nCheck).*
>
> While a bursty tracing framework captures temporal execution detail of frequently executed code paths, many program defects only manifest on rarely visited code regions that periodic bursty trace sampling alone is likely to miss. *This shortcoming can be overcome by a variation of the bursty tracing framework that includes adaptive instrumentation, where a sampling rate at which bursty traces of the instrumented code are sampled is adapted to the frequency of execution of the respective code path.*

Í[Application, at page 5, line 17 to page 6, line 9; emphasis added.] Applicants have previously noted examples of adaptation of sampling in the Application.

Wu cannot teach or suggest the above-quoted language of claim 1 because Wu directly instruments code in a profiled program and creates no duplicate code paths. The Action, in its rejection, cites to step 210 and accompanying text of Figure 2 as well as column 4, line 64 to column 5, line 5 of Wu. [*See*, Action at § 12, page 12.] The text accompanying step 210 of Figure 2 simply states "[a]t block 210, the compiler software inserts or modifies profiling instructions into the program and arranges the profile data." [Wu, at column 5, lines 15-17.] In

the other cited passage, Wu provides additional details about how this is adding of instrumentation is done indiscriminately:

> A low overhead collaborative profiling technique for collecting edge frequency profiles continuously is disclosed. To collect edge frequency profile, the prior art profiling technique insert a load, an increment, and a store into each edge that needs profiling. *In the present profiling technique, the compiler passes profiling requests as a few bits in branch instructions to the hardware, and the hardware executes the profiling operations with minimum impact on user program execution.*

[Wu, at column 4 line 64 to column 5, line 5.] As the quoted portions demonstrate, Wu's profiling techniques utilize modifications that are performed directly on the code of the program being tested. Wu does not appear to discuss either a) creating duplicate code paths, or b) instrumenting only the duplicate code paths.

Wu cannot teach or suggest adapting sampling rate for code paths because Wu only checks if an execution edge is profiled after it is already executed. Figures 5A and 5B of Wu, which are cited in the rejection of claim 1, each perform a "Was profiling instruction executed?" decision step. [*See*, Wu, at Figures 5A, step 510 and 5B, step 560.] As the language accompanying these steps makes clear, this no control is made over whether a step is a profiled step or not. For example:

> Flow continues to decision block 510. If a profiling instruction is executed, flow continues to processing block 515. If the profiling instruction is not executed, flow loops back to decision block 510, until a profiling instruction is executed. Once a profiling instruction is executed, the profile counters are updated at processing block 515.

[Wu, at column 8, lines 58-63, describing Figure 5.] A similar passage can be found describing Figure 5B. As such Wu is clear that it only observes whether profiling instructions are executed and does not change code paths in the middle of execution. Indeed, Wu would not have alternative code paths to switch between, as it does not create duplicate code paths.

Figures 5A and 5B do, as the Action notes, utilize various counters to determine frequency of code execution. However, as Wu makes clear, these counters are implemented in hardware to gain profile information and to determine when a profile phase transition should occur:

> The profiling hardware 400 signals a profile phase transition by generating an interrupt when the pir.trigger_counter reaches zero.

[Wu, at column 8, lines 30-32.] . The phase transition, however, does not modify the execution of the profiled program by adapting a sampling rate between the code paths. Rather it causes a complete re-optimiztion of the program itself:

> Flow continues to decision block 530, where *if trigger_counter reaches zero, an interrupt is generated at processing block 535 which signals a phase transition.* Flow continues to processing block 540 where new phase transition information is generated. *At processing block 545, the dynamic optimizer may take over and use the new edge profile information to re-optimize the program.* Flow then returns to start block 501.

[Wu, at column 9, lines 15-20.] Thus, while the program may be re-optimized, Wu clearly describes this re-optimization as the result of the tripping of the trigger_counter, and rather than an adaptation of a sampling rate between non-existent alternative code paths. Applicants also do not believe that the Action's suggested modification of Wu, which would determine a rate based on profile information, would suffice to teach or suggest the above language of claim 1, as it once again simply provides for a re-optimization trigger and does not create or utilize duplicate code paths.

For at least these reasons, Wu does not teach or suggest the above-recited language of claim 1 and thus does not describe each and every element of claim 1. Claim 1 is thus allowable and applicants request their allowance.

Claim 3

Claim 3, as amended, recites, in part:

> *providing a duplicate version of at least some already present procedures in the program with instrumentation for capturing runtime program data*;
> executing the duplicate version of at least some of the procedures; and
> *subsequently, for each of the already present procedures, selectively reducing a frequency at which the duplicate version of the procedure is executed,* the frequency equivalent to the number of executions of the duplicate version taken as a percentage of the total number of executions of the procedure.

[Emphasis added.] In its rejection of claim 3, the Action cites to similar passages of Wu as in the rejection of claim 1. [*See*, Action at § 12, page 14-15.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 3 and thus does not describe each and every element of claim 3. Claim 3 is thus allowable and applicants request their allowance.

Claim 6

Claim 6, as amended, recites, in part:

> *creating a copy of at least some of the original procedures in the computer program;*
> inserting instrumentation into the copies;
> creating an executable version of the program containing the original procedures and the copies; and
> executing the executable version of the program, wherein the copies of the procedures are executed in bursts, and *wherein a frequency at which the bursts are performed decreases as the total number of executions of both the original procedure and the copy of the procedure is executed,* said frequency at which the bursts are performed comprising the number of executions of the copy of the procedure divided by the total number of executions of both the original procedure and the copy of the procedure.

[Emphasis added.] In its rejection of the above-emphasized language of claim 6, the Action cites to similar passages of Wu as in the rejection of claim 1. [*See*, Action at § 12, page 16.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 6 and thus does not describe each and every element of claim 6. Claim 6 is thus allowable and applicants request its allowance.

Claim 14

Claim 14 recites, in part:

> *creating an instrumented version of the software containing an original version and an instrumented version of at least some procedures in the software, wherein the instrumented versions comprise instrumentation points;*
>
> . . .
>
> *executing the instrumented version of the software, wherein the additional programming code is executed more frequently when located at instrumentation points for procedures that are less frequently executed, and the additional programming code is executed less frequently when located at instrumentation points for procedures that are more frequently executed;*

[Emphasis added.] In its rejection of the above-emphasized language of claim 14, the Action cites to similar passages of Wu as in the rejection of claim 1. [*See*, Action at § 12, page 17.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 14 and thus does not describe each and every element of claim 14. Claim 14 is thus allowable and applicants request its allowance.

Claim 18

Claim 18 recites, in part:

> *producing a copy* of at least some procedures of the software;
> inserting instrumentation into the copies; and
> *sampling a copy of a procedure at a rate inversely proportional to how frequently both the original and the copy of the procedure are executed*;

[Emphasis added.] In its rejection of claim 18, the Action cites to similar passages of Wu as in the rejection of claim 1. [*See*, Action at § 12, page 19.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 18 and thus does not describe each and every element of claim 18. Claim 18 is thus allowable and applicants request its allowance.

Claim 23

Claim 23 recites, in part:

> *producing a copy of at least some procedures of the software*;
> inserting instrumentation into the copies; and
> *sampling a copy of a procedure at higher rates for procedures whose original versions [[or]] and copies are executed less frequently and sampling a copy of a procedure at lower rates for procedures whose original versions [[or]] and copies are executed more frequently . . .*

[Emphasis added.] In its rejection of claim 23, the Action cites to similar passages of Wu as in the rejection of claim 1. [*See*, Action at § 12, page 20.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 23 and thus does not describe each and every element of claim 23. Claim 23 is thus allowable and applicants request its allowance.

Claim 7

Claim 7, as amended, recites, in part:

> *creating an instrumented version of the software containing an original version and an instrumented copy version of each procedure in the software;*
> executing the instrumented version of the software, *wherein the instrumented version of the procedures are sampled at higher rates for procedures whose total number of executions of both the original versions [[or]] and the copy versions copies are executed less frequently and sampled at lower rates for procedures whose total number of executions of both the original versions [[or]] and the copy versions copies are executed more frequently . . . .*

[Emphasis added.] The Action rejects claim 7 over Wu in view of Alexander. In its rejection of claim 7, the Action cites to similar passages of Wu as in the rejection of claim 1. [*See*, Action at § 12, page 20.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 7 and thus does not describe each and every element of claim 7. Applicants further do not find relevant disclosure in Alexander, which is focused on detection of memory leaks in object-oriented environments through particular storage of execution metrics. Therefore, neither Wu nor Alexander, taken either separately or in combination, teach or suggest each limiation of claim 7. Claim 7, as well as claims 8-13, which depend from claim 7 are thus allowable and applicants request its allowance.

Dependent Claims

The Action rejects claims 2, 4, and 5 over Wu. Each of claims 2, 4, and 5 depend from claims 1 and 3 respectively and recite additional patentable language. In the interest of expediency, Applicants do not belabor the individual language of each claim but note that, for at least the reasons given above, Wu fails to teach or suggest each and every element of these dependent claims.

The Action rejects claims 8-10, 12, 13 and 15-17 over Wu in view of Alexander. Each of claims 8-10, 12, 13, and 15-17 depend from claims 7 and 14 respectively and recite additional patentable language. In the interest of expediency, Applicants do not belabor the individual language of each claim but note that, for at least the reasons given above, Wu and Alexander, taken either separately or in combination fail to teach or suggest each and every element of these dependent claims.

The Action rejects claims 20-22, 24, and 25 over Wu in view of Zorn. Each of claims 20-22, 24, and 25 depend from claims 18, and 23, respectively and recited additional patentable language. In the interest of expediency, Applicants do not belabor the individual language of each claim but note that, for at least the reasons given above, Wu fails to teach or suggest each and every element of these dependent claims. Applicants further do not find relevant disclosure in Zorn, which is focused on recording memory allocation for C and Lisp functions. As such, Wu and Zorn, taken either separately or in combination fail to teach or suggest each and every element of these dependent claims.

For at least these reasons, the rejection of claims 2, 4, 5, 8-10, 12, 13, 15-17, 20-22, 24, and 25 over the combinations of Wu and Alexander or Wu and Zorn is improper and fails to establish prima facie. Thus, Applicants respectfully note that the claims are allowable. Applicants respectfully request their allowance.

Interview Request

If the claims are not found by the Examiner to be allowable, the Examiner is requested to call the undersigned attorney to set up an interview to discuss this application.

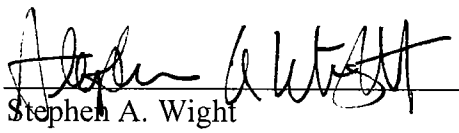
Conclusion

The claims in their present form should be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By 
Stephen A. Wight
Registration No. 37,759